ECE 375 PRELAB FOR LAB 4

Data Manipulation & the LCD

Lab Time: Friday 4 - 6

Faaiq Waqar

PreLab Questions

To complete this prelab, you may find it useful to look at the AVR Starter Guide and the AVR Instruction Set Manual. If you consult any online sources to help answer the pre lab questions, you must list them as references in your prelab.

1) What is the stack pointer? How is the stack pointer used, and how do you initialize it? Provide pseudocode (not actual assembly code) that illustrates how to initialize the stack pointer

The stack pointer, notated by SP, is used to point to the top of the stack. This is used to implement the LIFO (last in first out) data structure. The stack here is primarily used for storing temporary data, such as when used for subroutine calls in order to call back to the last return placement. The stack pointer is also used for return addresses after interrupts. The register itself will always point to the top of the stack. The implementation of the AVR stack pointer works as 2 8-bit special function registers, SPH and SPL. In order to initialize the stack pointer, low and high bits are assigned to SPL and SPH respectively using the OUT AVR command, used for IO ports, along with LDI to transfer bits to registers that can use the OUT command into the SPH and SPL registers.

Pseudocode:

Load Immediate R16 <- Low Stack Bits, Output into Stack Pointer Low SPL <- R16

Load Immediate R16 <- High Stack Bits, Output into Stack Pointer High SPH <- R16

2) What does the AVR instruction LPM do, and how do you use it? Provide pseudocode (not actual assembly code) that shows how to setup and use the LPM instruction.

LPM refers to the AVR instruction Load Program Memory. This allows the users to load data into registers from program flash memory. LPM can be used to access either 8-bit or 16-bit constants stored in the program memory, as the program flash is 16-bits wide. LPM works with the address register Z, and can be used in three ways, all of which will be represented in the pseudocode. LPM has post increment abilities, making it easy to index through arrays, and can also be used with no other parameters, where the usage is implied to R0 from Z.

Pseudocode:

Load Program Memory ; This will load Z memory into register R0

Load Program Memory R5 Z ; This will load Z memory into register R5

Load Program Memory R5 Z+ ;This will load Z memory in register R5 and post increment Z

3) Take a look at the definition file m128def.inc (This file can be found in the Solution Explorer → Dependencies folder in Atmel Studio, assuming you have an Assembler project open and you have already built an assembly program that includes this definition file. Two good examples of such a project would be your Lab 1 and Lab 3 projects.) What is contained within this definition file? What are some of the benefits of using a definition file like this? Please be specific, and give a couple examples if possible.

The definition file essentially contains lots of EQU and DEF directives, as well as other useful information such as the last addresses in Data Memory (RAMEND). This file is really useful because despite overlap of devices having a status register, or other parts of memory that are crucial but don't have a definite location, the definition file can predefine the utilities needed to operate using the device. Giving predefined locations as set by the file can allow the programmer to more easily utilize functionality of the device without having to memorize specific locations or struggle with hardware transition